**INDEX**

|  |  |  |  |
| --- | --- | --- | --- |
| No |  | Title | Page.No |
| 1 |  | **Introduction** |  |
|  | 1.1 | Problem Definition |  |
|  | 1.2 | Need for the New System |  |
|  | 1.3 | Project Scope |  |
| 2 |  | **Analysis** |  |
|  | 2.1 | Feasibility Study   * Technical Feasibility * Economical Feasibility * Operational Feasibility |  |
|  | 2.2 | Hardware and Software Requirements |  |
| 3 |  | **Design** |  |
|  | 3.1 | ER Diagram |  |
|  | 3.2 | Database Design / Data Dictionary |  |
|  | 3.3 | DFD Diagram |  |
| 4 |  | **Screenshots and Code** |  |
| 5 |  | **Limitations** |  |
| 6 |  | **Future Enhancement** |  |
| 7 |  | **References and Bibliography** |  |

**1. INTRODUCTION**

The Stock Portfolio Tracker is a Python-based program which is used for managing and analyzing their stock portfolios. Having a tool to monitor investments, simulate trading methods, and evaluate portfolio performance is essential in the world of financial markets. This initiative seeks to offer a flexible and approachable alternative for people looking to learn more about their stock ownership.

Through the yfinance library, the program makes use of the Yahoo Finance API's strength to make it simple for users to access real-time stock data. Portfolios can be created, customized, and chosen by users for a unique investing experience. By using a text-based menu system, user interaction is made simpler and is therefore usable by both novice and seasoned investors.

The Stock Portfolio Tracker meets a variety of user demands with features like adding and removing stocks, calculating total portfolio value, and running backtesting simulations. With the help of the backtesting tool, users may assess the previous performance of their portfolios based on various trading methods, giving them important information for upcoming investment choices.

The project makes use of the concepts of object-oriented programming, with classes like serving as the framework for the software. PyQt5 will be used by the software to create an accessible user interface for the user.

This program can be used to make investment decisions in accordance with various investment techniques. It will soon feature an option to trade automatically, eliminating the need for you to continuously keep an eye on your portfolio during market hours.

**1.1 PROBLEM DEFINITION**

Individuals must effectively manage their stock portfolios in the constantly changing financial markets while minimizing expenses related to data acquisition, analysis tools, and trial-and-error tactics. The cost-effectiveness of portfolio management for regular investors is hampered by the existing options, which frequently have high entry barriers.

The Stock Portfolio Tracker tries to address this financial situation by providing a simple and affordable solution. Real-time stock data and sophisticated analytical tools are frequently subject to hefty fees on traditional financial data platforms, which limits access for many people. This project fills the requirement for a cost-effective substitute by enabling users to make use of the Yahoo Finance API through the free and open-source yfinance library.

The Stock Portfolio Tracker avoids the need for pricey subscriptions and offers a user-friendly interface for managing stock portfolios with a focus on simplicity and cost savings. The project guarantees a cost-effective yet robust solution for customers to monitor, analyze, and optimize their investments without breaking the bank by utilizing Python's adaptability and integrating modular components.

By broadening financial insights, the Stock Portfolio Tracker not only addresses the cost issues with portfolio management but also makes them available to a wider audience. This initiative supports the objective of giving people access to a low-cost tool that enables them to successfully navigate the complexity of the stock market, promoting wise decision-making and financial success.

**1.2 NEED FOR THE NEW SYSTEM**

The inherent restrictions and difficulties that investors have while managing their stock portfolios are what drove the development of the Stock Portfolio Tracker. Existing systems frequently fail to satisfy users' changing needs, which forces the creation of a fresh, cutting-edge solution.

1. Cost-Efficiency: Modern financial platforms and portfolio management tools frequently have hefty subscription costs, which restrict access to important financial data and analytical functions. The Stock Portfolio Tracker satisfies this demand for cost-effectiveness by making use of open-source libraries and offering consumers a cost-effective alternative to managing their portfolios.
2. Accessibility: A steep learning curve for individual users is caused by the complexity and institutional investors-focused design of many existing systems. The new system is built with accessibility in mind and has an easy user interface (UI) for users with different levels of experience. This guarantees that investors of all experience levels can use the program with ease.
3. Data Accuracy and Real-Time Insights: In the quick-paced world of stock trading, accurate and timely information is essential. In order to give customers access to the most recent information for efficient decision-making, the new system makes use of the Yahoo Finance API to provide real-time stock data.
4. Investors frequently lack the tools necessary to test and improve their trading methods. Backtesting for Informed methods. With the introduction of the backtesting tool in the Stock Portfolio Tracker, users can now assess previous portfolio performance based on various methods. This capability supports strategy optimization and helps with informed decision-making.

In summary, the Stock Portfolio Tracker provides a system that is affordable, easily accessible, adaptable, and user-friendly in order to meet the urgent needs of contemporary investors. This initiative seeks to democratize financial tools and equip users to successfully navigate the intricacies of the stock market by aligning with the changing needs of ordinary investors.

**1.3 PROJECT SCOPE**

The Stock Portfolio Tracker has comprehensive features and functionalities that it brings to users / investors seeking a simplified yet powerful tool for managing and optimizing their stock portfolios. The project encompasses the following key aspects:

1. **Portfolio Creation and Management:**
   * Users can create multiple portfolios for their investment goals.
   * Portfolios can be customized by adding, removing, or adjusting stock holdings.
2. **Cost-Efficient Data Access:**
   * The system uses the Yahoo Finance API through the yfinance library, eliminating the need for subscriptions.
   * Real-time stock data is accessible to users without incurring additional expenses.
3. **User-Friendly Interface:**
   * The UI provides an intuitive interface for users of all experience levels.
   * Navigating through portfolio is easy
4. **Backtesting Strategies:**
   * The system includes a backtesting feature, allowing users to evaluate the historical performance of their portfolios based on different trading strategies.
   * Backtesting results provide valuable insights for refining and optimizing investment approaches.
5. **Accessibility and Inclusivity:**
   * The project caters to a broad audience by prioritizing accessibility and inclusivity.
   * Novice investors can easily navigate the system, while experienced users benefit from advanced features**.**

**2. ANALYSIS**

**2.1 FEASIBILITY STUDY**

2.1.1 **Technical Feasibility**

The Stock Portfolio Tracker's technical viability is evaluated to make sure that the system's implementation is in line with the technology and resources available. By leveraging modules like yfinance for stock data retrieval and pyqt5 for the graphical user interface, the project makes advantage of Python's powerful capabilities. The scalability and maintainability of the system are facilitated by these technologies, which are widely supported. The selection of SQLite for data storage guarantees effective data management and retrieval.

The adoption of object-oriented programming (OOP) principles also makes it easier to organize and modularize code. The system is adaptive to changing requirements because of the system's technical architecture, which makes adding new features and upgrades simple.

2.1.2 **Economical Feasibility**

The Stock Portfolio Tracker's development and upkeep from a financial standpoint are assessed in the economical feasibility study. In terms of software development, Python's open-source nature and the availability of free, widely-used libraries help keep costs down. By using SQLite as the database management system, expensive database licenses are not required.

Future improvements can be made without making substantial financial commitments thanks to the project's scalability. The majority of the system's features can be used without incurring additional expenditures, while there may be operational costs related to data subscriptions for real-time stock data.

2.1.3 **Operational Feasibility**

Operational feasibility evaluates how easily the Stock Portfolio Tracker may be used and implemented in the operational environment that is envisioned. PyQt5 is used in the system's user-friendly interface design for simple navigation. To offer a seamless user experience, the functionalities—such as adding stocks, buying and selling, and portfolio analysis—are intuitively linked.

The capacity to analyze financial data is improved with the addition of elements like data visualization using matplotlib. By utilizing SQLite for local data storage, the system may be used offline, enhancing accessibility and lowering reliance on external services.

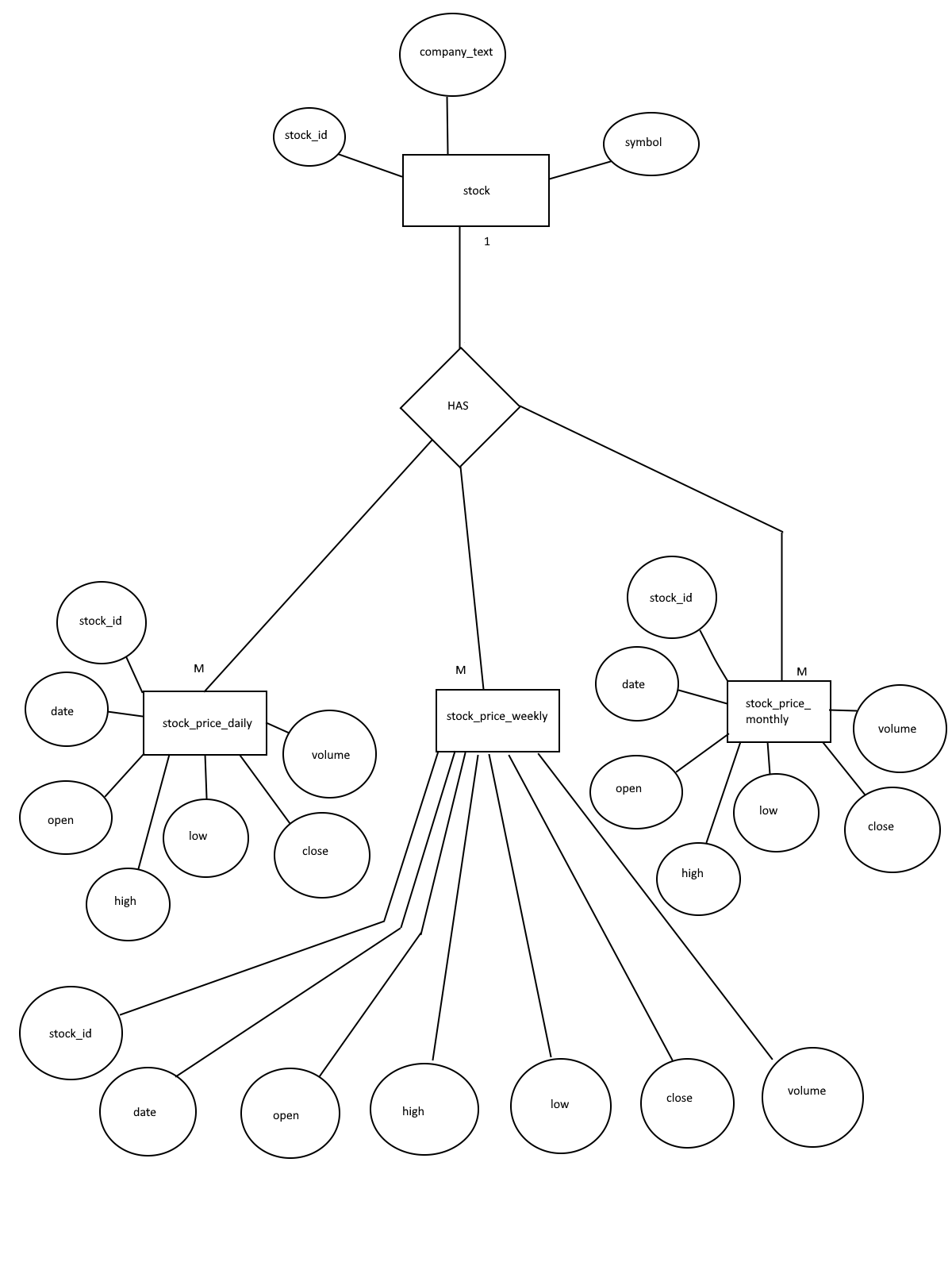
The Stock Portfolio Tracker is a practical and efficient tool for managing and assessing stock portfolios since it exhibits strong technical, financial, and operational feasibility.

**2.2 HARDWARE REQUIREMENTS**

1. **Processor (CPU):**
   * Any modern multi-core processor.
   * Dual-core or higher recommended for optimal performance during data processing.
2. **Memory (RAM):**
   * Minimum of 4 GB RAM.
   * 8 GB or higher recommended for handling larger datasets and smoother multitasking.
3. **Storage:**
   * At least 50 MB of available storage space for the application and its dependencies.
   * Additional storage space may be required based on the size of the portfolio data.
4. **Display:**
   * A monitor with a resolution of 1280x720 or higher.
   * Higher resolution displays enhance the visualization of charts and graphs.
5. **Input Devices:**
   * Keyboard and mouse for data input and interaction with the graphical user interface.
   * Touchscreen devices are supported but not required.
6. **Internet Connection:**
   * While the application can operate offline, an internet connection is recommended for fetching real-time stock data.
7. **Operating System:**
   * Compatible with Windows, macOS, or Linux operating systems.
8. **Graphics Card:**
   * Basic integrated graphics are sufficient.
   * A dedicated graphics card is not necessary for standard functionality.

**3. DESIGN**

3.1 ER Diagram

sdads

3.2 Data Dictionary

Table: stock

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P/F | Field Name | Caption | Data Type | Field Size | Notes |
| P | stock\_id |  | INT |  |  |
|  | symbol |  | TEXT |  |  |
|  | company\_text |  | TEXT |  |  |

Table : stock\_price\_daily

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P/F | Field Name | Caption | Data Type | Field Size | Notes |
| F | stock\_id | daily | INT |  |  |
|  | date |  | DATE |  |  |
|  | open |  | REAL |  |  |
|  | high |  | REAL |  |  |
|  | low |  | REAL |  |  |
|  | close |  | REAL |  |  |
|  | volume |  | REAL |  |  |

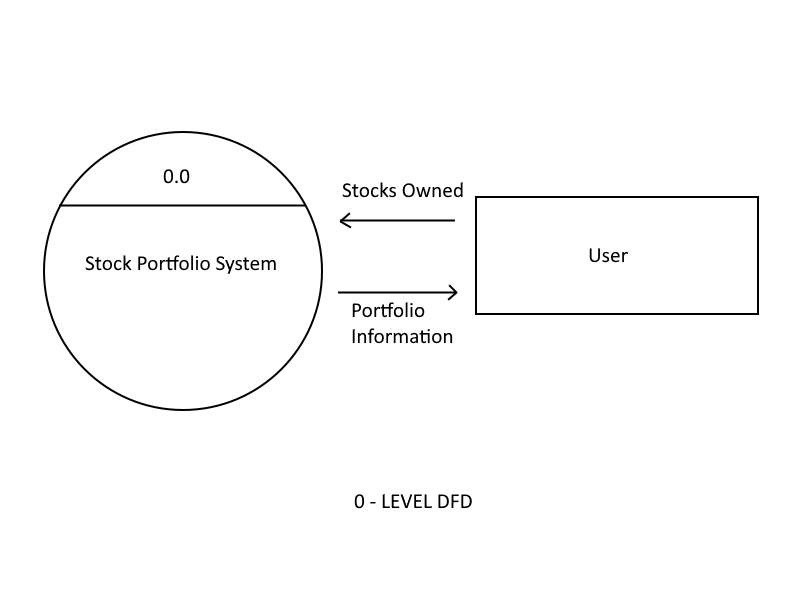
Table : stock\_price\_weekly

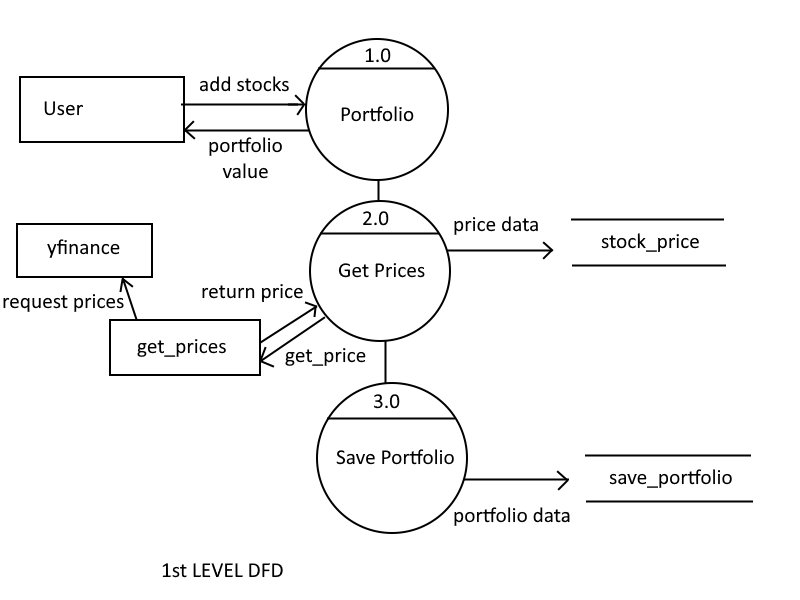
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P/F | Field Name | Caption | Data Type | Field Size | Notes |
| F | stock\_id | weekly | INT |  |  |
|  | date |  | DATE |  |  |
|  | open |  | REAL |  |  |
|  | high |  | REAL |  |  |
|  | low |  | REAL |  |  |
|  | close |  | REAL |  |  |
|  | volume |  | REAL |  |  |

Table : stock\_price\_monthly

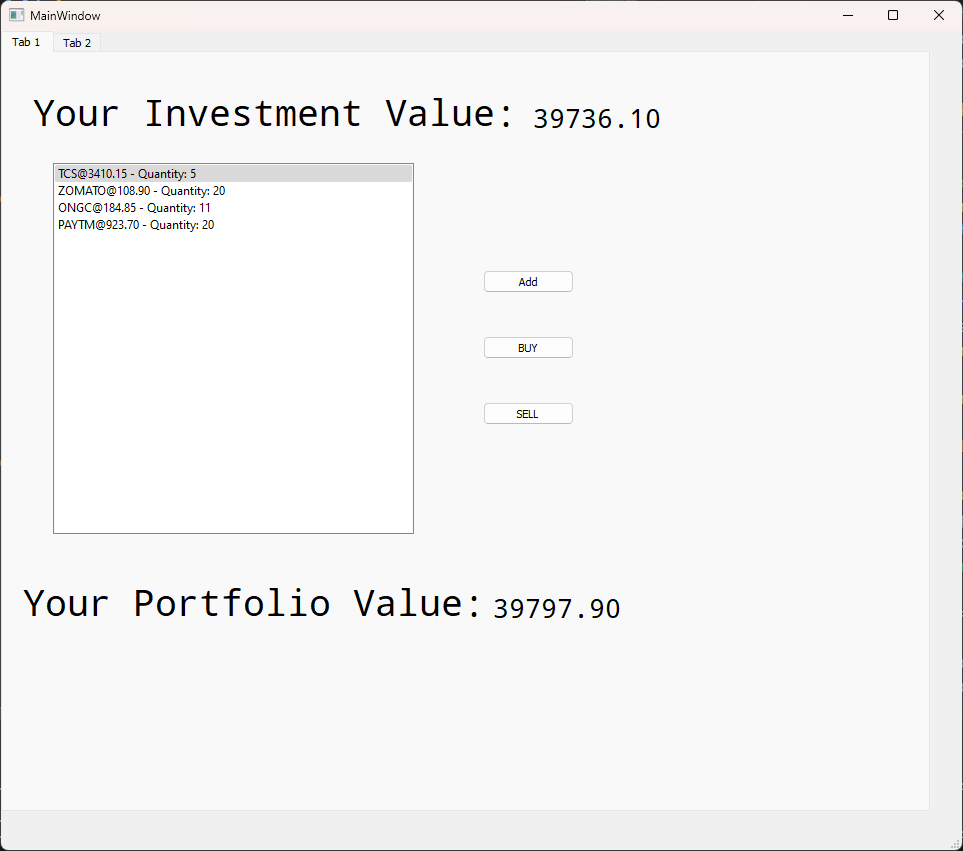
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P/F | Field Name | Caption | Data Type | Field Size | Notes |
| F | stock\_id | Monthly | INT |  |  |
|  | date |  | DATE |  |  |
|  | open |  | REAL |  |  |
|  | high |  | REAL |  |  |
|  | low |  | REAL |  |  |
|  | close |  | REAL |  |  |
|  | volume |  | REAL |  |  |

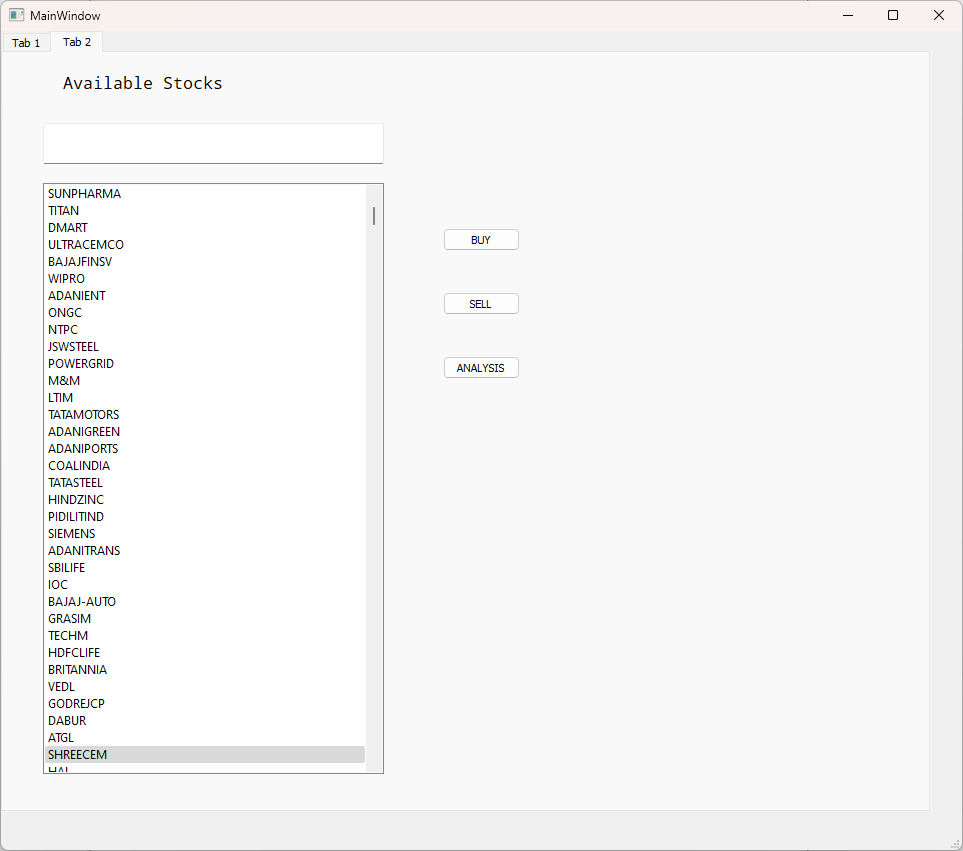
3.3 DFD





**4. SCREENSHOTS AND CODE**

****

****

****

**5. LIMITATIONS**

**1. Reliance on outside APIs**

For stock data, the project uses external APIs. These APIs are subject to change, and any disruptions could impact the project's functionality. Some APIs may also have usage restrictions or subscription fees for extensive use. Since this API is free, its not as performant.

**2. .Simple Investment Techniques:**

The project might concentrate on fundamental investment tactics. Option trading and algorithmic trading are two more sophisticated tactics that might not be completely supported.

**3. Users' Level of Skill**

The project presupposes that users have a certain amount of financial literacy. It may be difficult for new investors or those who are not experienced with the markets to evaluate the data and come to wise decisions.

**4. Scalability**

The project may encounter difficulties managing a sizable user base or sizable data volumes. For the best performance, scalability issues must be taken into account.

**6. FUTURE ENHANCEMENTS**

**1. Real-Time Data Updates:**

Implement real-time data updates to provide users with the most current stock prices. Utilize WebSocket connections or other technologies for live updates.

**2. Advanced Analytics:**

Enhance the analytics capabilities with advanced charting tools, technical indicators, and performance metrics. Provide users with insights into portfolio volatility, beta, and other advanced analytics.

**3. Risk Management Features:**

Integrate risk management tools to help users assess and manage the risk associated with their portfolios. This could include features such as value-at-risk (VaR) calculations.

**4. Integration with Zerodha API**

Integration with Zerodha's API allows users to utilize advanced order types offered by the brokerage, such as limit orders, market orders, and more. This ensures flexibility and precision in trade execution.

**5. Algorithmic trading**

Algorithmic trading, also known as algo trading, introduces automated trading strategies based on predefined rules and algorithms. Integrating algorithmic trading features into the Stock Portfolio Tracker enhances the platform in several ways.

**7. REFERENCES & BIBLIOGRAPHY**

1. geeksforgeeks.org

2. docs.python.org

3. tutorialspoint.com

4. qt.io

5. reddit.com

6. stackoverflow.com